```
//[木塊問題/The Blocks Problem]
#define IN "P0505IN.txt"
#define OUT "P0505OUT.txt"
//***************************************
#include <iostream>
#include <time.h>
using namespace std;
void redir(void);
//***************************************
/* Work Space*/
#include <string>
#include <vector>

void find_block(int a, int &pa, int &ha);
void clear_above(int p, int h);
void pipe_over(int p, int h, int p2);
void show(void);

int n;
vector<int> pipe[25]; //二維陣列(第一維大小根據題意, 第二維大小不固定)
//***************************************
int main(void)
{
    redir(); //redirection
//***************************************
/* Work Space*/
    int i;
    string s1, s2;
    int a, b;
    int pa, pb; //pa, pb: position of a and b
    int ha, hb; //ha, hb: height of a and b

    cin >> n;
    for(i=0; i<n; i++){
        pipe[i].push_back(i);
    }

    while(1){
        cin >> s1;
        if(s1 == "quit"){
            break;
        }else{
            cin >> a >> s2 >> b;

            find_block(a, pa, ha);
            find_block(b, pb, hb);

            if(pa == pb){
                continue; //非法指令
            }
            if(s1 == "move"){
                clear_above(pa, ha);
            }

            if(s2 == "onto"){
                clear_above(pb, hb);
            }

            pipe_over(pa, ha, pb);
        }
    }
    show();
//***************************************
```

```
    freopen("CON", "r", stdin); //取消重新導向
    freopen("CON", "w", stdout);

    printf("Time used = %.2f\n", (double)clock()/CLK_TCK); //傳回程式目前為止執行的時間

    system("pause");
    return 0; //the end...
}

void redir(void)
{
    freopen(IN, "r", stdin);
    freopen(OUT, "w", stdout);
}
//****************************************
/* Work Space*/
void find_block(int a, int &p, int &h)
{
    for(p=0; p<n; p++){
        for(h=0; h<pipe[p].size(); h++){
            if(pipe[p][h] == a){
                return;
            }
        }
    }
}

//把第p堆高度為h的木塊"上方的"所有木塊移回原位
void clear_above(int p, int h)
{
    int i;
    int b;

    for(i=pipe[p].size()-1; i>h; i--){
        b = pipe[p][i];
        pipe[b].push_back(b);
    }
    pipe[p].resize(h+1);
}

//把第p堆高度h"及其上方"的木塊整體移動到p2堆的頂部
void pipe_over(int p, int h, int p2)
{
    int i;

    for(i=h; i<pipe[p].size(); i++){
        pipe[p2].push_back(pipe[p][i]);
    }
    pipe[p].resize(h);
}

//輸出結果
void show(void)
{
    int i, j;

    for(i=0; i<n; i++){
        printf("%d: ", i);
        for(j=0; j<pipe[i].size(); j++){
            printf("%d ", pipe[i][j]);
        }
        printf("\n");
    }
```

```
}

//Input(IN) Sample
/*
10
move 9 onto 1
move 8 over 1
move 7 over 1
move 6 over 1
pile 8 over 6
pile 8 over 5
move 2 over 1
move 4 over 9
quit
*/
```